Known limitations
for {log} version 4.6.14

Interactive environment

Language features (general)

- In RUQs and in Intensional Set terms the control term must be a variable (not any term). For example,
  \[ S = \{ [X,Y]: X \text{ in } (1) \land Y \text{ in } (2) \} \]
is not allowed.

- Evaluable compound integer expressions can occur only in the built-in binary predicates \(\text{is}, =<, <, \geq, >, =:=, =\backslash=\)
  like in Prolog. However, differently from Prolog, they can contain uninstantiated variables. For example,
  \[ X \text{ in } \text{int}(1,10) \land X \text{ is } Y + 1 \]
is allowed, while
  - \[ X \text{ in } \text{int}(1,10) \land f(X) = f(Y + 1) \rightarrow \text{no} \]
  - \[ p(X,X+\!1) :- q(X) \]
  - \[ (2 + 1) \text{ in } \text{int}(1,4) \rightarrow \text{no} \]
  - \[ \text{integer}(1+2) \rightarrow \text{no} \]
  are not allowed.

- (Mixing set and interval terms) Set terms of the form \(\{t_1,\ldots,t_n\backslash\text{int}(a,b)\}\) are not allowed in user programs and goals. However, they are dealt with correctly within the interpreter and they can be returned as the result of a set constraint. For example:
  \[
  \text{log}=> X \text{ in } \{0\backslash\text{int}(1,10)\}.
  \text{wrong set term}
  \]
  \[
  \{\text{log}\}=> \text{un}(\text{int}(1,2),\text{int}(3,4),R).
  \]
  \[
  R = \{1,2\backslash\text{int}(3,4)\}
  \]

Constraint solver

- Interval bounds must be instantiated to integer constants (no uninstantiated variables nor compound integer expressions are allowed). For example:
  \[
  \{\text{log}\}=> X \text{ in } \text{int}(A,6).
  \text{INSTANTIATION ERROR: interval bounds must be known values}
  \]
  \[
  \{\text{log}\}=> X \text{ in } \text{int}(1+2,6).
  \text{no}
  \]
  \[
  \{\text{log}\}=> 1 \text{ in } \text{int}(0,B) \land B = 3.
  \text{INSTANTIATION ERROR: interval bounds must be known values}
  \]
  \[
  \{\text{log}\}=> B = 3 \land 1 \text{ in } \text{int}(0,B).
  B = 3
  \]

- (Constraints over lists) The constraint solver is not able to prove that the following constraint involving lists is unsatisfiable:
  \[
  \{\text{log}\}=> X \text{ in } L \land L \text{ in } X \land \text{list}(L).
  \text{true}
  \]
Constraint: X in L, L in X, list(X), list(L)

- (FD solver incompleteness) \{log\} uses an incomplete FD solver, hence it can be not able to detect unsatisfiability of some constraints involving integer variables. Providing a finite domain for the integer variables possibly occurring in the constraint guarantees completeness of the solver. For example:

\[ \{\text{log}\} \rightarrow X > Y \land Y < X. \text{true} \]
Constraint: integer(Y), integer(X)

\[ \{\text{log}\} \rightarrow \text{un}(X,Y,Z) \land \text{size}(X,NX) \land \text{size}(Z,NZ) \land NZ<NX. \text{true} \]
Constraint: un(X, Y, Z), size(Y, _G7609), _G7609 in int(0, sup), size(X, NX), size(Z, NZ), set(Y), set(X), NX in int(3, sup), set(Z), NZ in int(2, sup)

Adding a finite domain for \(X\) and \(NX\), respectively, we get the (desired) failure (also with automatic labeling disabled)

\[ \{\text{log}\} \rightarrow X > Y \land X < Y \land X \in \text{int}(1,10). \text{no} \]

\[ \{\text{log}\} \rightarrow \text{un}(X,Y,Z) \land \text{size}(X,NX) \land \text{size}(Z,NZ) \land NZ<NX \land NX \in \text{int}(0,10). \text{no} \]